Vulnerabilities That Threaten Web Applications in Afghanistan

Sayed Mansoor Rahimy, Sayed Hassan Adelyar, and Said Rahim Manandoy

Abstract Familiarizing web developers with different types of vulnerabilities lead to the creation of secure web applications. In the last few decades, there has been considerable interest in web hacking which leads to different types of web attacks that can cause financial damages, privacy loss, data loss, and life-threatening situations. This study aims to discover the most common web vulnerabilities that exist in Afghanistan's web applications and websites. We conducted this study by using Netsparker, Skipfish, and Acunetix web vulnerability scanners with the standard web vulnerability assessment (WVA) method. The result shows that almost all the web applications in Afghanistan are vulnerable to different types of cyber-attacks. A total of 997 instances of various types of vulnerabilities were detected on 109 web applications from three different domains. This study presents 24 common vulnerabilities, which is more than prior studies. The results of this study familiarize web developers with the most common vulnerabilities that can exist in a typical web application. Therefore, this study will encourage them to consider these vulnerabilities during the software development life cycle.

Keywords Cyber-attacks · Vulnerability assessment · Web vulnerability scanners · Web application

1 Introduction

Web applications become one of the most dominant technologies for delivering dynamic services over the Internet. They are used for delivering various types of services such as e-government, financial, social, and Learning Management Systems (LMSs). For this reason, a huge amount of sensitive data is exchanged through web applications. Web applications are cross-platform, responsive, interactive, remotely accessible, fast deployable, and user-friendly. The web platform is composed of

S. M. Rahimy (✉) · S. H. Adelyar · S. R. Manandoy
Salam University, Kabul, Afghanistan
e-mail: s.mansoor@salam.edu.af

C. Onwubiko et al. (eds.), *Proceedings of the International Conference on Cybersecurity, Situational Awareness and Social Media*, Springer Proceedings in Complexity, https://doi.org/10.1007/978-981-99-6974-6_12

different parts. The web server provides web application services. Web clients access the web application via HTTP protocol in a web browser. A wide range of technologies are available for web application development.

However, widespread adoption has led them to face numerous security threats due to their complex infrastructure for hosting and deployment coupled with diverse development technologies available on both server side (e.g., PHP or ASP) and client side (e.g., HTML, CSS JavaScript, or Flash). As a result, developing and deploying a secure web application is rigid [13]. Hereby, web security is a vital component to be considered by all organizations over the world.

Here in Afghanistan, most web designers and developers focus on product functionality and Quality of Experience (QoE) rather than security requirements and best practices. Lack of security awareness and experts in the organizations in Afghanistan lead to a wide range of security breaches in their web applications. Lack of due diligence in most web users from cyber-attacks causes them to lose their sensitive information and even threaten their life. Consequently, we assume that a high number of web vulnerabilities exist in most of the web applications in Afghanistan [7, 12, 18].

To the best of our knowledge, there are insufficient studies in the literature regarding vulnerability assessment and security of web applications in Afghanistan. Therefore, the present work aims to discover the most common web vulnerabilities in mostly used Afghani web applications. The results of this study provide web designers and developers with a productive method for web vulnerability assessment at minimum cost, time, and effort. Moreover, it will help them to be aware of some common web vulnerabilities. The following are the three most important research questions, which are answered in this paper:

What are the most common web vulnerabilities in Afghani web applications?

Which security vulnerability assessment method is suitable for identifying these vulnerabilities?

What are the threats that are associated with the most commonly detected vulnerabilities?

The remaining parts of the paper are structured as follows. Section 2 provides related works and additional information relevant to the vulnerabilities of web applications. Section 3 presents the methodology and tools used for the vulnerability assessment of web applications. Section 4 presents the result of the vulnerability assessment. Section 5 presents a detailed discussion and implications. Finally, Sect. 6 concludes this paper by summarizing the research work, giving the contributions achieved, and showing directions for future work.

2 Literature Review

In this era of digitalization, most businesses are relying on web applications. Service providers use web applications to communicate with their subscribers. Therefore, web applications become interesting targets for most of the attackers on the Internet.

In the following subsections, we briefly review some of the existing literature regarding web application security vulnerability assessment and web vulnerability scanners.

2.1 Web Application Vulnerability Assessment

Vulnerability assessment is a proactive approach through which we can identify and scan for the existing vulnerabilities in the system before they could be exposed by attackers with malicious intents [6, 22]. There are different methods for identifying vulnerabilities. Static analysis, attack graph analysis, and vulnerability scanners are the most well-known methods for vulnerability assessment. Static analysis analyzes the structure of the program and the code of the program to detect flaws. Some techniques used in static analysis are lexical analysis, type reference, constraint analysis, and many more. Attack graph analysis represents all the paths followed by attackers to achieve their desired goals. This method is used for identifying vulnerabilities inside a network. For instance, some techniques which are used for generating the attack graphs are clustered adjacency matrix, hierarchical aggregation, minimization analysis, ranking graphs, and game theoretics. Vulnerability scanners are software tools used to identify vulnerabilities in a network system and/or in a software application. There are different vulnerability scanners used for network vulnerability scanning, operating system vulnerability scanning, and web vulnerability scanning [6].

Farah et al. performed black-box testing in (2018) to identify XSS and CSRF vulnerabilities in 500 Bangladeshi web applications. This study showed that 30% of Bangladeshi web applications are vulnerable to XSS and CSRF attacks. Of 500 web applications, 335 of them were found vulnerable to either XSS or CSRF or both attacks. Their results have shown that about 65% of the 335 web applications were vulnerable to XSS attacks and 75% of them were vulnerable to CSRF attacks.

Moniruzzaman et al. conducted black-box and white-box testing research [11] on identifying common vulnerabilities in Bangladeshi websites. This study aimed to represent a framework for identifying maximum vulnerabilities at minimum cost and effort. They considered six different attack vectors which are SQLi, XSS, BAS, CSRF, Unusual Ports, and Deprecated TLS. Black-box testing was conducted with the help of Kali Linux penetration testing tools and white-box testing was conducted with the help of static code analysis techniques. In this study, they found that 36% of the websites in Bangladesh are secure and 64% of them are running with various vulnerabilities.

Ahmed and Murah analyzed the security of 16 Libyan governmental websites [1]. This study proposed a safety classification matrix for the 16 websites using 4 safety categories: safe, somewhat unsafe, unsafe, and highly unsafe. To classify a website in one of these safety levels, they first assessed the website for vulnerability using Netsparker and Acunetix. Secondly, they determined whether sensitive information is encrypted or not during transactions. Finally, they evaluated SSL encryption using the Qualys SSL Labs tool. They analyzed one website as highly unsafe, six websites as unsafe, eight websites as somewhat unsafe, and one website as safe.

Nirmal K. et al. stated in [14] that it is very critical to hardened web applications due to their existence in various businesses. This paper focuses on performing vulnerability assessment and penetration testing during various phases of the Software Development Life Cycle (SDLC). Security considerations and best practices should be embedded in each phase of the web application development life cycle.

Sri Devi and Kumar executed a vulnerability analysis on 100 websites in [21]. This study used Nikto and OWASP Zed Attack Proxy (ZAP) vulnerability scanners and provides a comparison of these scanners. The study shows that both vulnerability scanners identify various vulnerabilities. Some vulnerabilities are detected by the Nikto but not by the ZAP and vice versa. Nikto provides some additional information such as server, SSL information, and ciphers.

Trapti Jain and Nakul Jain conducted an experimental study on implementing multithreading concepts using multiple vulnerability scanners [10]. They used Python as a scripting engine in which Whatweb, Nikto, Dirb, and Nmap are executed parallel. Furthermore, they performed data normalization and parsing techniques on the result of the scanners and stored them in a database. Lastly, they used ModSecurity WAF to mitigate the discovered vulnerabilities by implementing custom configuration rules. This study shows that running multiple scanners at the same time reduces the execution time and provides an effective result.

2.2 Web Application Vulnerability Scanners

Web vulnerability scanners are used to identify various flaws and weaknesses such as misconfigurations, outdated files, and common vulnerabilities that can be found in a web application. Various web vulnerability scanners are available in the market. Some of the most well-known web vulnerability scanners are Netsparker, Acunetix, Nessus, Nikto, Skipfish, Dirbuster, Burp Suite, Vega, OpenVAS, ZAP Proxy, Sqlmap, W3af, Xsser, and many more.

Antunes and Vieira conducted a comparative study of penetration testing tools and static code analyzers on the detection of SQLi in a set of web services [5]. Three commercial web penetration testing tools: HP WebInspect, IBM Rational AppScan, and Acunetix WVS compared with three static code analyzers: FindBugs, Yasca, and IntelliJ IDEA against eight web services. The result has shown that static code analyzers can detect more vulnerabilities than penetration testing tools and have better coverage. On the other hand, static code analyzers have a high rate of false positives than penetration testing tools.

Bairwa et al. have conducted a comparative study on five vulnerability scanners in [6]. Their observation has shown that different scanners identify different types of vulnerability but a single tool is not capable of detecting all types of vulnerabilities. They identified the capability of each vulnerability scanner by running each one of them against several web applications. They highlighted that Nessus is the only scanner that has detected most of the vulnerabilities followed by Acunetix and Burp Suite.

Qianqian and Xiangjun discussed in [16] about open-source vulnerability scanners. By comparing these open-source scanners, they select the W3af vulnerability scanner for enhancing the capability of identifying Clickjacking vulnerability in HTML5 pages. They made a custom script and used it in an actual test, the result has shown that vulnerability can be detected.

Rajan and Erturk conducted a case study on Acunetix WVS (web vulnerability scanner) in [17]. In this study, they focused on how important it is to scan web applications for their vulnerabilities with the help of WVSs. This study has shown that WVSs help to speed up the web applications' vulnerability scanning process.

Patel and Gosavi present a vulnerability scanning system architecture based on HTTP methods [15]. The ingredients of the system are URL Crawling, Domain Reputation, CMS Scan, URL Scan, Search Engine, Remote Site, and 3rd Party Databases. The system provides the following features: 1—BackdoorWebShellLocator; 2—domain reputation in Google, SURBL, Malware Patrol, Clean-Mx, and Phistank; 3—Mail Server IP Check-in 58 repositories; 4—Scan SQL Injections for MySQL, MSSQL, PGSQL, and Oracle databases; 5—Scan XSS; 6—Scan Malware; 7—Detect and Scan CMS; 8—Scan for Directory Indexing.

Wang et al. studied the detection technology of common application vulnerabilities and the way vulnerability scanning tools work [22]. This paper designed and implemented the vulnerability scanning system for Web Applications of Power Company. The system is scalable and can scan multiple target websites at the same time.

Huang et al. introduced a new vulnerability scanner, VulScan in [9]. VulScan automatically generates test data and can discover injection and cross-site scripting (XSS) vulnerabilities by using penetration testing and evasion techniques. This study also proposes three main categories of countermeasures for mitigating SQL injections and XSS attacks. The countermeasures are secure implementation, defense mechanism deployment, and penetration testing.

Alzahrani et al. conducted a comparative study on ten different web vulnerability scanners [3]. Netcraft is a tool that can be used for web server fingerprinting. It can be used by attackers to gather information about web servers, underlying operating systems, server uptime, and much more information. SSLyze, Qualys SSL Labs, and OpenSSL are the tools that can be used for vulnerability detection in the transport layer. XSS Server, XSSer, and Xenotix XSS are the scanning tools used for XXS vulnerability detection. SQL Inject-ME, SQLninja, and Havij are the tools that can be used for the detection of SQLi vulnerabilities. A comparison of some of the well-known web vulnerability scanners that has been conducted by [2, 8] are presented in Table 1.

Table 1 Comparison between well-known web vulnerability scanners

WVS	Attack vector support (%)	Automated crawling (WIVET score) (%)	Detection accuracy of backup/ hidden file (%)	Detection accuracy of RFI (%)	Detection accuracy of reflected XSS (%)	Detection accuracy of SQLi (%)
Burp Suite Professional	19	96.00	25.00	72.22	96.97	100
IBM AppScan	30	92.00	5.43	100	100	100
NTOSpider	19	94.00	42.00	79.63	100	97.06
Tinfoil Security	19	94.00	100	100	100	100
WebInspect	29	96.00	100	100	100	100
ZAP	17	73.00	38.04	100	100	
Netsparker	30	92.00	100	100	100	100
Acunetix	25	94.00	32.61	77.78	100	100

3 Methodology

An empirical research study was conducted based on a standard vulnerability assessment method described in the literature [4, 8, 20]. In brief, this section reviews the method with some minor modifications. The vulnerability assessment was conducted on 109 web applications from three top-level governmental (.gov.af), educational (.edu.af), and commercial (.com.af) domains. The vulnerability assessment was carried out in four steps as shown in Fig. 1.

3.1 Step One—Reconnaissance

It was carried out to identify the targets and gather as much information as possible from targeted web applications. The black-box testing techniques are used in this step. Tools that were used for this step are the Whois, Dig, Nslookup, theHarvester, Robtex, and Netcraft. A brief description of these tools is provided in Table 2.

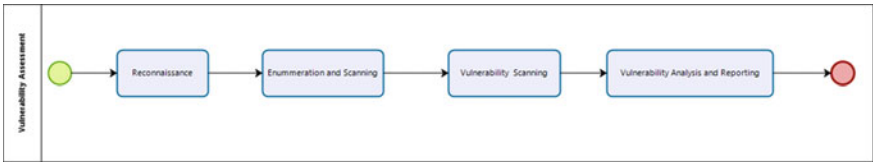
Vulnerability Assessment
Reconnaissance
Enummeration and Scanning
Vulnerability Scanning
Vulnerability Analysis and Reporting

Fig. 1 Vulnerability assessment method

Table 2 Reconnaissance tools

Name	Description	Environment
Whois	Extracts domain information	Kali Linux
Dig	Provides DNS information	Kali Linux
Nslookup	Provides DNS interrogation and zone transfer services	Kali Linux
Robtex	Provides DNS information	https://www.robtex.com
Netcraft	provides a web server and web hosting market-share analysis	https://www.netcraft.com
theHarvester	Searches for domains in various search engines	Kali Linux

Table 3 Enumerations and scanning tools

Name	Description	Environment
Nmap	Utility for network discovery and security auditing	Kali Linux
Recon-ng	Web reconnaissance framework is written in Python	Kali Linux
Knockpy	Python for DNS brute forcing	Kali Linux
Netcat	Used for banner grabbing	Kali Linux

3.2 Step Two—Enumeration and Scanning

The purpose of this step is to enumerate and scan for information such as the web server, underlying operating system, virtual host environment, load balancers, and proxies. The tools that were used in this step are the nmap, recon-ng, Knockpy, and Netcat. A brief description of these tools is provided in Table 3.

3.3 Step Three—Vulnerability Scanning

This step was carried out to find vulnerabilities in the target web applications. Using a single vulnerability scanner can lead to false positive results. Therefore, three different web vulnerability scanners were used. The Netsparker and Acunetix are commercial and have a nice graphical user interface. Skipfish is a free command line tool available in Kali Linux. Table 4 presents the environments in which these scanners are installed.

Table 4 Vulnerability scanners

Name	Description	Environment
Netsparker	Commercial web vulnerability scanner	Windows 10
Acunetix	Commercial web vulnerability scanner	Windows 10
Skipfish	Free web vulnerability scanner	Kali Linux

3.4 Step Four—Vulnerability Analysis and Reporting

This step provides an in-depth analysis and statistics of the detected web vulnerabilities. Furthermore, detailed descriptions of the vulnerabilities are presented and reported to all governmental, educational, and commercial organizations.

4 Result

In this study, a total of 109 web applications from three different domains were selected in the reconnaissance phase. Of 109 web applications, 62 of them are governmental, 39 of them are educational, and 8 of them are commercial. Figure 2 shows the percentage of these three different types of web applications for various domains. Significantly, 997 instances of different types of vulnerabilities were detected by Netsparker, Skipfish, and Acunetix WVSs in the vulnerability scanning phase. These WVSs classified vulnerabilities according to CVE and CVSS into High, Medium, Low, and Informational levels. From 997 instances of vulnerabilities, 86 instances are High level, 167 instances are Medium level, 311 instances are Low level, and 433 instances are Informational level. Vulnerabilities that are in informational level are not real vulnerability rather they help an attacker to further investigation. Figure 3 presents the dominance of these levels in percentages using a pie chart.

In this study, we found 55 different forms of vulnerabilities. Among them, some of them have a high frequency. Table 5 presents some common of them that exist across multiple web applications.

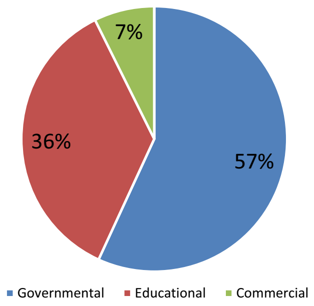

Fig. 2 Percentages of web applications

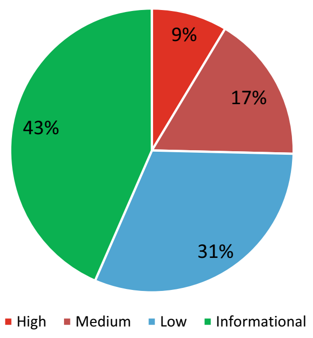

Fig. 3 Percentages of vulnerabilities

5 Discussions

This study aimed to discover the most common web vulnerabilities in the most commonly visited web applications in Afghanistan. The results highly supported our hypothesis, and we found that all 109 web applications vulnerable to one or more cyber-attacks. It was assumed that high numbers of web vulnerabilities exist in most of the web applications in Afghanistan. While not all of the vulnerabilities were high level, the overall results present a large number of vulnerabilities across multiple web applications in various domains.

In 2020, Naier et al. conducted security testing on 135 websites under the.af domain using the OWASP risk rating methodology. They have shown that 92% of the websites had below 3 high-risk level vulnerabilities, 13% of the websites had below 3 medium-risk level vulnerabilities, most of the websites had low-risk level vulnerabilities, and 60% of websites had information-risk level vulnerabilities. In this study, we have used a more standard and accurate methodology. In addition, the sample size in our study is smaller than their study, but we have gathered more vulnerabilities. This indicates that our results are more accurate than them. We have discovered 997 instances of web vulnerability by Netsparker, Acunetix, and Skipfish WVSs.

A similar study is conducted by Ali and Murah in 2018. They discovered a total of 522 instances of different types of vulnerabilities in 16 Libyan governmental websites using Netsparker and Acunetix WVSs. We discovered a total of 997 instances of different types of vulnerabilities on 109 web applications using Netsparker, Skipfish, and Acunetix WVSs. Thus, the number of vulnerabilities in Libyan governmental websites is more than the Afghani websites. However, Ali and Murah presented 9 common web vulnerabilities in 16 Libyan governmental websites and we presented 24 almost different common web vulnerabilities in 109 web applications.

With the existence of these vulnerabilities in web applications, a huge set of threats can be launched. Here, we use STRIDE threat modeling to categorize these vulnerabilities [19]. For example, the existence of SQL Injection vulnerability in a web application could cause a massive financial loss (tampering). Table 6 shows

Table 5 Common web vulnerabilities

Vulnerability	Level of severity	Instances	Number of web applications
Cross-site scripting	High	73	5
SQL injection	High	3	1
Microsoft IIS tilde directory enumeration	High	6	2
Long password denial of service	High	1	1
Elasticsearch service accessible	High	1	1
Vulnerable JavaScript libraries	Medium	48	27
TLS 1.0 enabled	Medium	24	24
User credentials are sent in clear text	Medium	12	12
Slow HTTP denial of service attack	Medium	11	11
Source code disclosures	Medium	8	8
TLS/SSL Sweet32 attack	Medium	7	7
TLS/SSL weak cipher suites	Medium	7	7
Clickjacking: X-Frame-Options header missing	Low	60	60
Unencrypted connection	Low	45	45
HSTS not implemented	Low	32	32
Cookies with missing, inconsistent, or contradictory properties	Low	31	31
Login page password-guessing attack	Low	26	26
Cookies without the HttpOnly flag set	Low	24	24
Cookies without secure flag set	Low	23	23
Insecure referrer policy	Informational	76	76
Content Security Policy (CSP) not implemented	Informational	74	71
No HTTP redirection	Informational	38	38
Outdated JavaScript libraries	Informational	38	23
Subresource Integrity (SRI) not implemented	Informational	30	27

vulnerabilities and their related threats. Furthermore, additional study is required to find measures and mitigation techniques to defend against these threats.

As mentioned in the Introduction, most web designers and developers focus on product functionality rather than security considerations. The results of this study will familiarize and encourage them to take some security considerations during the software development life cycle.

Table 6 Associated threats with common detected vulnerabilities

Vulnerability	Threats
Cross-site scripting	Tampering
SQL injection	Tampering
Microsoft IIS tilde directory enumeration	Information disclosure
Long password denial of service	DoS
Elasticsearch service accessible	Information disclosure
Vulnerable JavaScript libraries	One or more
TLS 1.0 enabled	Information disclosure
User credentials are sent in clear text	Spoofing
Slow HTTP denial of service attack	DoS
Source code disclosures	Information disclosure
TLS/SSL Sweet32 attack	Information disclosure
TLS/SSL weak cipher suites	One or more
Clickjacking: X-Frame-Options header missing	Spoofing, information disclosure, and elevation of privileges
Unencrypted connection	Information disclosure
HSTS not implemented	Information disclosure
Cookies with missing, inconsistent, or contradictory properties	Elevation of privileges
Login page password-guessing attack	Spoofing
Cookies without the HttpOnly flag set	Tampering
Cookies without secure flag set	Information disclosure
Insecure referrer policy	Information disclosure
Content Security Policy (CSP) not implemented	Tampering
No HTTP redirection	Information disclosure
Outdated JavaScript libraries	One or more
Sub Resource Integrity (SRI) not implemented	Spoofing

6 Conclusion

A vulnerability assessment was completed against 109 web applications and a total of 997 different instances of web vulnerabilities were discovered. Additionally, a detailed description of these vulnerabilities was reported and presented to all related organizations. Furthermore, a statistical analysis of these vulnerabilities has been presented using pie charts and tables. The result of this study will help web designers and developers to build secure web applications. Moreover, the results of the identified vulnerabilities will be shared with their corresponding web application owners to be addressed soon. Although our results are limited to Afghanistan web applications. However, it remains to be further clarified whether our results could be applied to other web applications in other countries. Future work will mainly cover the presentation of countermeasures that will help to mitigate these web vulnerabilities.

References

1. Ahmed Ali, A., Murah, M.Z.: Security assessment of libyan government websites. In: 2018 Cyber Resilience Conference (CRC). IEEE (2018)
2. Alsaleh, M., Alomar, N., Alshreef, M., Alarifi, A., Al-Salman, A.M.: Performance-based comparative assessment of open source web vulnerability scanners. Secur. Commun. Netw. (2017)
3. Alzahrani, A., Alqazzaz, A., Fu, H., Almashfi, N., Zhu, Y.: Web application security tools analysis. In: 2017 IEEE 3rd International Conference on Big Data Security on Cloud (2017)
4. Ansari, J.A.: Web Penetration Testing with Kali Linux: Build Your Defense Against Web Attacks with Kali Linus 2.0. Packt Publishing (2015)
5. Antunes, N., Vieira, M.: Comparing the effectiveness of penetration testing and static code analysis on the detection of SQL injection vulnerabilities in web services. In: 2009 15th IEEE Pacific Rim International Symposium on Dependable Computing, PRDC 2009, pp. 301–306 (2009)
6. Bairwa, S., Mewara, B., Gajrani, J.: Vulnerability scanners: a proactive approach to assess web application security. Int. J. Comput. Sci. Appl. **4**(1), 113–124 (2014)
7. Dilipraj, E.: South Asian cyber security environment: an analytical perspective centre for air power studies. In: Asian Defence Review, pp. 161–190. Knowledge World Publishers (2014)
8. Felderer, M., Büchler, M., Johns, M., Brucker, A.D., Breu, R., Pretschner, A.: Security testing: a survey. In: Advances in Computers, vol. 101, pp. 1–51. Web Application Vulnerability Scanner & Sap Security Tools. Academic Press Inc. (2023). https://piter0ff.wordpress.com/web-application-vulnerability-scanner-sap-security-tools/
9. Huang, H.-C., Zhang, Z.-K., Cheng, H.-W., Shieh, W.S.: Web Application Security—Threats Countermeasures and Pitfalls. IEEE Computer Society (2017)
10. Jain, T., Jain, N.: Framework for web application vulnerability discovery and mitigation by customizing rules through ModSecurity. In: 2019 6th International Conference on Signal Processing and Integrated Networks (SPIN) (2019)
11. Moniruzzaman, M., Chowdhury, F., Ferdous, M.S.: Measuring vulnerabilities of Bangladeshi websites. In: 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE). IEEE (2019)
12. Naier, M.M., Hamidi, A., Momand, R.: Analysis of Web Application Security Vulnerabilities: A Case Study of Web Applications in Afghanistan, vol. 4 (2020)
13. Nath, H.V.: Vulnerability assessment methods—a review. CCIS **196**, 1–10 (2011)

14. Nirmal, K., Janet, B., Kumar, R.: Web application vulnerabilities—the hacker's treasure. In: 2018 International Conference on Inventive Research in Computing Applications (ICIRCA) (2018)
15. Patil, H.P., Gosavi, P.B.: Web vulnerability scanner by using HTTP method. Int. J. Comput. Sci. Mob. Comput. **4**(9), 255–260 (2015)
16. Qianqian, W., Xiangjun, L.: Research and design on web application vulnerability scanning service. In: 2014 IEEE 5th International Conference on Software Engineering and Service Science (2014)
17. Rajan, A., Erturk, E.: Web Vulnerability Scanners: A Case Study (2017)
18. Salamzada, K., Shukur, Z., Abu Bakar, M.: A framework for cybersecurity strategy for developing countries: case study of Afghanistan. Asia-Pacific J. Inf. Technol. Multimed. (2015)
19. Shostack, A.: Threat Modeling—Designing for Security. Wiley (2014)
20. Singh, H., Sharma, H.: Hands-on Web Penetration Testing with Metasploit—The Subtle Art of Using Metasploit 5.0 for Web Application Exploitation. Packt Publishing (2020)
21. Sri Devi, R., Mohan Kumar, M.: Testing for security weakness of web applications using ethical hacking. In: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184) (2020)
22. Wang, B., Liu, L., Li, F., Zhang, J., Chen, T., Zou, Z.: Research on web application security vulnerability scanning technology. In: 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC) (2019)